Timers

- New functionality in EJB 2.1
  - in past you were almost forced to do this from a client application!
- Useful for project
public abstract class MyClass
    implements javax.ejb.EntityBean, javax.ejb.TimedObject {
    javax.ejb.EntityContext ejbCtx;

    public void setEntityContext(javax.ejb.EntityContext ctx) {
        ejbCtx = ctx;
    }

    public void ScheduleWork(String desc, Date when) {
        TimerService tmrSvc = ejbCtx.getTimerService();
        tmrSvc.createTimer(when, desc);
    }

    public void ejbTimeout(javax.ejb.Timer timer) {
        // application logic for timeout
        ...
    }
TimerService Interface

- Access to creation / enumeration of timeout requests

```java
package javax.ejb;
public interface TimerService {
    public Timer createTimer(Date expiration, Serializable info) throws ... ;
    public Timer createTimer(long duration, Serializable info) ...;
    public Timer createTimer(
        Date initialExpiration, long intervalDuration,
        Serializable info) ...;
    public Timer createTimer(
        long initialDuration, long intervalDuration,
        Serializable info) ...;

    public java.util.Collection getTimers() ...;
}```
Timer Interface

- A (1-shot or interval) timeout request

```java
package javax.ejb;

public interface Timer {
    public void cancel() throws ...
    public java.io.Serializable getInfo() ...
    public java.util.Date getNextTimeout() ...
    public long getTimeRemaining() ...
    ...
}
```
More Details

- Identifying timers
  - the “info” can be any serializable object
- Cancelling
  - if it hasn’t expired yet
- Can ask for next expiration but that’s all
  - e.g. can’t tell if one-shot or periodic!
Transactions

• The `createTimer()` is transactional
  • in scope of current transaction
  • abort => timer not set

• Use txn attribute `RequiresNew` for the `ejbTimeout` method
  • runs in new container-initiated transaction
Entity Bean Timer

- Set timer per EJB class and primary key
  - when fires, implicit `findByPrimaryKey`
- removing an entity automatically cancels all timers for that PK
Session Bean Timer (Stateless)

- Set timer per EJB class
  - fires in *some* idle session EJB of the class
  - like a new anonymous session
- Timer in *stateful* session EJB does not make sense!
Message-Driven Bean Timer

- Expiration Date should be in message data
- The onMessage method sets the timeout
- The ejbTimeout method does the work

- As if message were timed
  - defer work from onMessage to ejbTimeout
  - eliminate duplicates in onMessage?
Weaknesses

- Timeout expressiveness
  - every weekday at midnite
  - first Tuesday of each month

- Subject to “implosions”
  - many timers set to expire simultaneously